CUBIT Capability Proposal

Technical Area Geometry, Meshing, Infrastructure, GUI, Graphics, etc..	**Technical Lead** Cubit Developer in charge of technical area
Parsing / Infrastructure	Darryl

MRD Description

Describe the capability in terms of how a user would see it.

None – users shouldn't see any change (except that some parsing errors should go away).

SRS Description

What needs to be done by Cubit developers to implement this capability? Break the tasks into steps if applicable. (Steps should be on the order of 2 man-weeks or more)

Design and implement a new parsing infrastructure. The new infrastructure should have the following characteristics:

1. Writing a single command syntax definition results in
    a. Automatic command recognition (developer doesn't have to write code to decide which command is being requested by a text string).
    b. Automatic use of abstractions where appropriate (such as parse_entities)
    c. Automatic generation of the command's help string
2. Commands can be specified independently. Changing the syntax or processing of one command should have no effect on the syntax or processing of another command.
3. When the parser recognizes a particular command, a specific function in the CUBIT API is called with appropriate parameter values filled in with the results of automatic parsing. The actual text of the command should never be required by the API function.
4. More robust syntax checking than we currently have.
5. Should support all our current parsing features (aprepro, partial word completion, syntax-based command line help)

Tasks include:

1. Identify a toolset for the job (flex/bison, ANTLR, Spirit?).
2. Design the new parsing infrastructure.
3. Implement the infrastructure
4. Enter syntax for existing commands into new system.
5. Hook up new commands to CUBIT API (see API proposal).

Justification

Describe why this is important and what impact it will have if it is implemented. (or not implemented).

Our current command handlers are fragile and intertwined. Changing code for one command often has side effects on several other commands. It is nearly impossible to determine all of the commands which use a particular piece of command handler code. There is currently no guarantee that the help string matches the correct command syntax. Most commands currently allow you to add junk to the end of the command and the command is still executed; the junk at the end is ignored without error or warning. Many commands also accept syntax errors in the middle of a command, with or without

warning. Some commands even reject correct syntax because of poor command handling practices, but it's difficult to fix because we can't determine the consequences of a change to the command handler without intimate understanding of the command in question AND every other command that may use the same command handling code (and since you can't figure out which commands share that section of code…). Our current command handlers currently mix a lot of command execution with command parsing, which leads to partial command execution and to unintentional changes in behavior when addressing parsing issues.
If a new parser is well designed, we should fix (or at least improve) all of the issues described above.
Note that this tasks depends on the existence of a CUBIT API, proposed separately. It should be possible to reverse the order of tasks so that the parsing infrastructure goes in first, followed by the development of the CUBIT API. We could migrate to the new parser one command at a time as they are made available via the API.

Resources Who will work on this	**Time estimate** How much time will it take in man-weeks	**Targeted Release** 10.2 (August 06), 10.3 (March 2007), 10.4 (August 2007), Future (beyond FY07)
Darryl	25 weeks	10.4

Submitted By:	**Date:**
Darryl	3/28/06